Convex Optimization For Machine Learning (cvx4ml)

Konstantin Burlachenko (bruziuz@stanford.edu, burlachenkok@gmail.com)

Problem (Motivation, 5 sentence)

"Humanity is a wandering fires in the fog. The appearance of breakthroughs through the fog from one flame to another can be called a miracle - A.N. Kolmogorov".
Machine Learning connects engineering fields with usual people life. But Machine Learning can be improved by mathematical optimization, which has already become an important tool in many areas.[2]

When I visited Stanford at 2NOV2017 I heard from prof. S.Boyd that this days our world have been fractured, even several years ago all people was happy with pure convex optimization. In this work I tried to make some steps to find interesting things in Convex Optimization which is usefull Machine Learning with the goal to "connect the fires".

One example is L_1 norm for feature selection, instead greedy approach:
E. Candes to S.Boyd: "L1 is least squares for 21 century" [1]

Covered and Improved Models from classical ML

Convex optimization is good for model fitting. But it's not only about that logistic regression, linear regression, lasso, SVM are reduced to it. After taking idea of convex optimization there are a lot of freedom in creating models in classical Machine Learning. Unfortunately it's approximately immediately close the doors to closed form solutions. General schema for fit data via loss function $\sum_{i=1}^{m} \mathcal{L}(x^{(i)}, y^{(i)}; \theta) + \lambda r(\theta)$ generates a lot of convex optimization problems - if $r(\theta)$ is convex and $\mathcal{L}(x^{(i)}, y^{(i)}; \theta)$ is convex in θ for fixed $x^{(i)}, y^{(i)}$.

I implemented solvers for several problems for classification:
1. Logistic regression, SVM, SVM without slab constraints
2. Variations of improved SVM

I showed extra things:
1. How extra constraints to linear regression can be exploited
2. How fit piecewise affine convex function into data
3. How improved L1 heuristic can be exploited

I implemented solvers for several problems for regression:
1. Dead zone fit and Log Barrier fit, Huber fir (Robust Linear regression)
2. Fit with L_1 norm, $L_1$with trust region, stochastic L_1
3. Fit with L_2 norm (least-square), L_2 with trust region, stochastic L_2
4. Fit with L_2 norm and L_1 norm as regularization (Lasso regression)
5. Fit with L_2 norm and L_2 norm as regularization (Ridge regression)
6. Fit with L_∞ (Chebyshev optimization problem), stochastic L_∞

My solvers are based on Interior-Point method. Nesterov and Nemirovski were the first to point out that interior-point methods can solve many convex optimization problem. It's not first-order. It doesn't use slow methods based on gradients or subgradients. Second order methods can be used by people with advanced Linear Algebra level, unfortunately. This is because Hessian structure should be exploited very carefully.

Datasets

For test my solvers and implementation which I created via using CVXPY and SkLearn I used following data, splitted by holdout cross-validation into 70%/30% train/dev set:
1. Regression Set-1, features from 3-14 and target variable is feature 2
   https://www.kaggle.com/harlfoxem/housesalesprediction/data
2. Classification Set-1, features 1-19, and target variable is feature 20
   https://www.kaggle.com/primaryobjects/voicegender/data

Results about model comparisions for this datasets

Model name	Accuracy on train	Accuracy on dev
binary_classification_logistic_regression	0.886175115207	0.27311827957
binary_classification_svm	0.928110599078	0.748387096774
binary_classification_svm with "20" reweighting iterations	0.920737327189	0.830107526882
binary_classification_svm_with_ellipsoid_surface	0.979723502304	0.852688172043
binary_classification_svm_without_slab_constraint	0.979262672811	0.851612903226
binary_classification_svm_without_slab_constraint	0.98064516129	0.866666666667
Number of total samples in train and dev: 3100		
Number of features (including intercept) 13		

Model name	Mean Square Error on Train	Mean Square Error on Dev
linear_regression_L2	1782.84852355	2739.76809487
linear_regression_L2_with_L1regularization, lamba=0.1	1808.69561332	2741.37679232
linear_regression_L2_with_L2regularization,lambda=0.2	1783.91515791	2732.96556962
linear_regression_L2_with_stochastich_robust, buckets=20	1783.05980256	2736.0088867
linear_regression_L2_with_trust_region, D=2000,x0=0	1802.60661679	2735.21353917
linear_regression_L2_with_worst_case , buckets=20	1790.75216496	2775.87643092
linear_regression_Linf	4202.59961688	7795.18892507
linear_regression_Linf_with_stochastich_robust,buckets=50	2958.29507905	4971.24011347
linear_regression_Huber, MHuber = 500.0	1852.24555136	2818.73484157
linear_regression_L1	1852.16000858	2818.41414265
linear_regression_L1_with_stochastich_robust , buckets=20	2111.77019395	3214.15590754
linear_regression_L1_with_trust_region,D=2000,x0=0	1872.52507745	2828.63282284
linear_regression_DeadZone	1852.16272262	2818.41181079

My implementation by experiments in Intel Core I-7, 2.7GHz, 8 GB RAM, x64 several times faster. then similar implementation in SkLearn or CVXPY. Details will be available in the final report.

Model name	Time for my sovler	Need time for other solver
linear_regression_L2	0.0820	0.172 (Cvxpy/MOSEK), [4]
linear_regression_L2_with_L2regularization	0.061	0.423(Cvxpy/ECOS), [4]
binary_classification_logistic_regression	0.01	0.049 (SkLearn), [10]

Discussion

Convex optimization relative to Machine Learning:
1. Give a way create models which are convex in parameters and find parameters.
2. Introduce new concepts – stochastic, worst case hard constraints to field of ML
3. Exploiting this concepts open a way to connect ML with existing people knowledge base via pull this into ML algorithms in form of constraint or via multiobjective optimization technics.
4. Give you a way combine several models sequentially – you start with one. And then formulate trust region around old solution and move to another solution.
5. Increase speed for existing models which are used in classical ML. Interior-point methods can solve optimization problem in a number of iterations that is almost always in the range between 10 and 100. Ignoring any structure in the problem (such as sparsity), each step requires on the order of $\max(n^3, n^2 m, F)$. Where: $F - \text{cost for compute gradients and hessians}, m - \text{number of contstaints}, n - \text{number of variables}$
6. It's very hard to say how things are changing. But ideas of gradients, subgradients, heavy-ball method variation is already in the field of interest of Machine Learning community. Even they was born in mathematic optimization. It is less well known that convex optimization has very effective algorithms that can reliably and efficiently solve even large convex problems beside optimization community.

Future Works

1. Find applications from convex optimization for ML based on SDP. E.g. Lowner-John (minimum volume) ellipsoid around train set find. Look into active points on surface. And make heuristic decision on is this points are outliers. Also one more application from convex optimization is estimation of inverse of covariance matrix for Gauss distribution under convex constraints.
2. Find applications of using SOCP for ML.
3. Build better models for Deep Learning based on non-convex optimization, which heuristically will give better solution. Right now only momentum and stochastic gradients are widely popularized, but there are exist other (Particle method, Difference of convex function, Sequential Convex Optimization, Branch and Bound). It's better to mention the quote:
   "In non-convex optimziation there a a lot of room for personal expression"-S.Boyd [3]

References

[1] Quote of Emmanuel Candes to S.Boyd https://youtu.be/DsXzUU691ts?t=3685
[2] Video for give IEEE James H. Mulligan, Jr. Education Medal for S.Boyd https://ieeetv.ieee.org/ieeetv-specials/stephen-p-boyd-accepts-the-ieee-james-h-mulligan-jr-education-medal-honors-ceremony-2017
[3] Quote "A lot of room for personal expression. Maybe your method works better for some problem", EE364b, 2008, L12,10:45 https://youtu.be/cHVpwyYU_LY?t=646
[4] CvxPy: http://www.cvxpy.org/en/latest/
[5] Convex Optimization 1st Edition by S.Boyd, L. Vandenberghe. https://www.amazon.com/Convex-Optimization-Stephen-Boyd/dp/0521833787
[6] Boris Polyak, Introduction to optimization https://www.amazon.com/Introduction-Optimization-Translations-Mathematics-Engineering/dp/0911575146
[7] CS229 lecture notes http://cs229.stanford.edu/syllabus.html
[8] EE364a: Convex Optimization I, S.Boyd, Stanford University http://stanford.edu/class/ee364a/
[9] EE364b: Convex Optimization II, S. Boyd, Stanford University https://stanford.edu/class/ee364b/
[10] Packet Scy-Learn (http://scikit-learn.org/stable/)